Putting ROSE To Work:

A Proposed Application of a Request-Oriented Scheduling Engine for Space Station Operations

Presented to SpaceOps 2000, Toulouse France, June 21, 2000

By
John Jaap & Kim Muery
Mission Support Systems Group, FD42
Ground Systems Department
Flight Projects Directorate
Marshall Space Flight Center

This presentation is the result of an on-going research project into advanced planning and scheduling systems.

Introduction

- **A Request-Oriented Scheduling Engine (ROSE) is one that processes a single request (adding a task to a timeline) and then waits for another request. Other scheduling engines are batch and user-assisting.**

- **It is possible to build and deploy a ROSE-based system that allows payload developers to schedule their own payloads.**

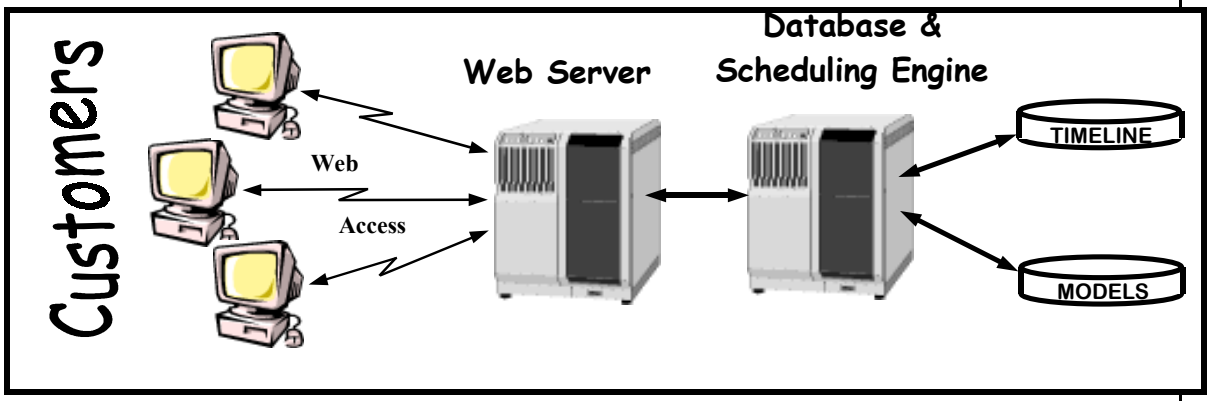
Two milestones must be accomplished to deploy such a system:

1. **The software system itself must be built.**
2. **An operations concept using the system must be designed and accepted.**

When payload developers produce their own timeline, they can get the timeline they want. They are the customers. Giving the customers what they want is the epitome of a better timeline.

A ROSE-based System

- **Web Application — The user navigates via a web browser to the ROSE web site, logs on, and proceeds to formulate scheduling requests and submit them for scheduling.**



- **Transaction-Quality Implementation — The system is sufficiently robust so that nothing is ever lost; hardware or software failures cannot cause the loss of what is already in the timeline.**

- **Graphics User Interface — The request formulation process (modeling) employs graphical methods to describe activities and sequences. The user is presented with a canvas to which items are added and arranged in a hierarchy (for activities) or in a network (for sequences). Details of the requirements are entered via dialog boxes.**

Graphical modeling is similar to the Interim User's Requirements Collection (iURC) system.

A ROSE-based System
The Critical Element

If payload developers, who already have expertise in the payloads, are to formulate scheduling requests and submit them to a scheduling engine, they must have expertise in the scheduling engine.

- **Being an expert in the behavior of the scheduling engine means knowing how the engine will react to a given model, and how to build a model to achieve the desired results.**
- **ROSE will make the payload developers virtual experts in using the scheduling engine by making them experts in modeling. ROSE uses a request format that is a natural representation of the requirements without adding artificial constraints or constructs — the model looks like the real-world payload and the engine interprets the model as expected.**
- **ROSE provides immediate feedback (when a request is submitted, the resulting timeline is available immediately), thereby exposing users to the workings of the scheduling engine.**

After only a few submittals, the users will know what to expect from the engine when a request is submitted; i.e., they will be *virtual experts.*

A ROSE-based System Equipment Modeling

In the ROSE modeling approach, nearly all resources are defined in the equipment models.

- **Payload Developers provide description of their equipment and how they intend to use it. They also provide descriptions of how they intend to use Station-provided equipment.**
- **Personnel at the Control Center (not the payload developers) build equipment models. These models define how much of each station resource is used when the equipment is operated. Most equipment has multiple operation modes with differing resource requirements; each of these modes is modeled.**
- **Equipment models include both information received from the payload developers and requirements derived from how the equipment is connected (integrated) with the station.**

Crew is treated as a unique resource, not as equipment. Additionally, an equipment model never specifies any crew usage.

A ROSE-based Operations Concept

- **The proposal presented in this paper is based on an instance of ROSE installed at a major support facility, such as the U.S. Payload Control Center, and serving all of the payload developers (PDs) supported by that facility. The payload developers would remotely connect to the ROSE system and simultaneously schedule their payloads. The results would be the timeline provided by that installation (or partner) to the Payload Operations Integration Function.**

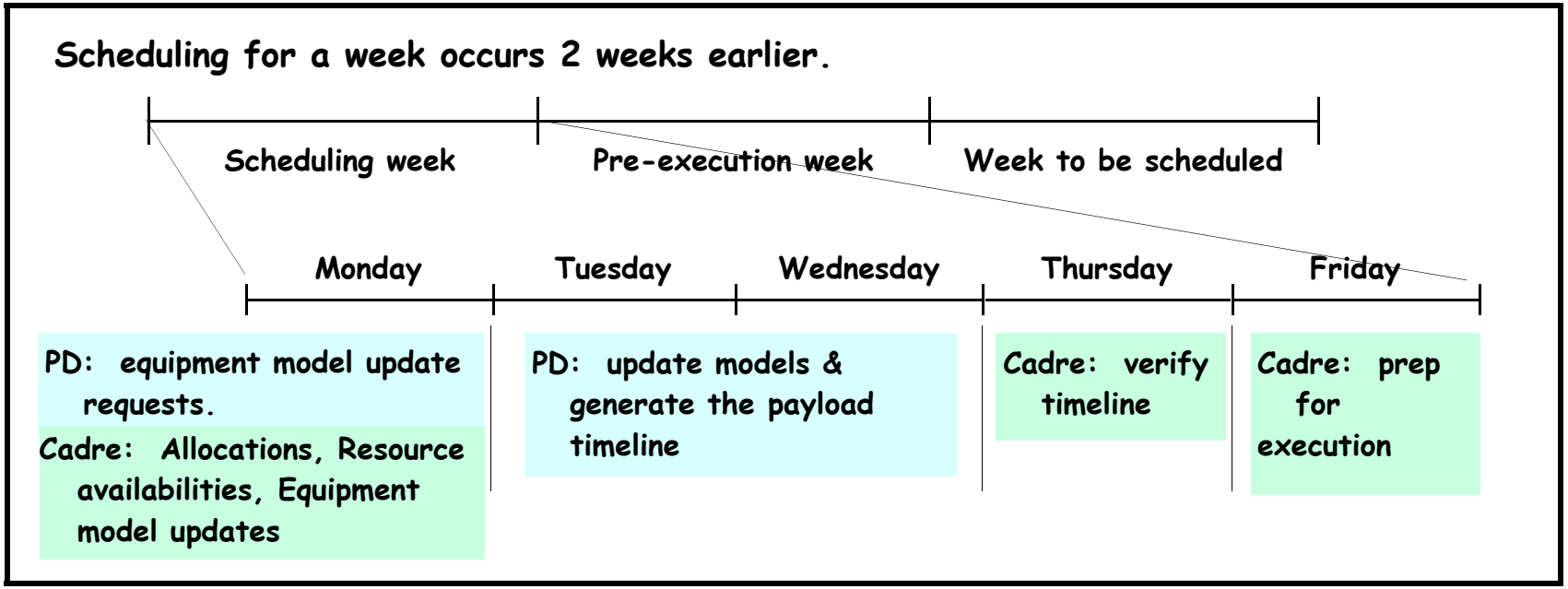
The operations concept proposed in this presentation is one of several ROSE-based operations concepts that are being discussed in conjunction with the ROSE research project.

The ROSE system and ROSE-based operations concepts can be used for scheduling applications other than the Space Station.

A ROSE-based Operations Concept

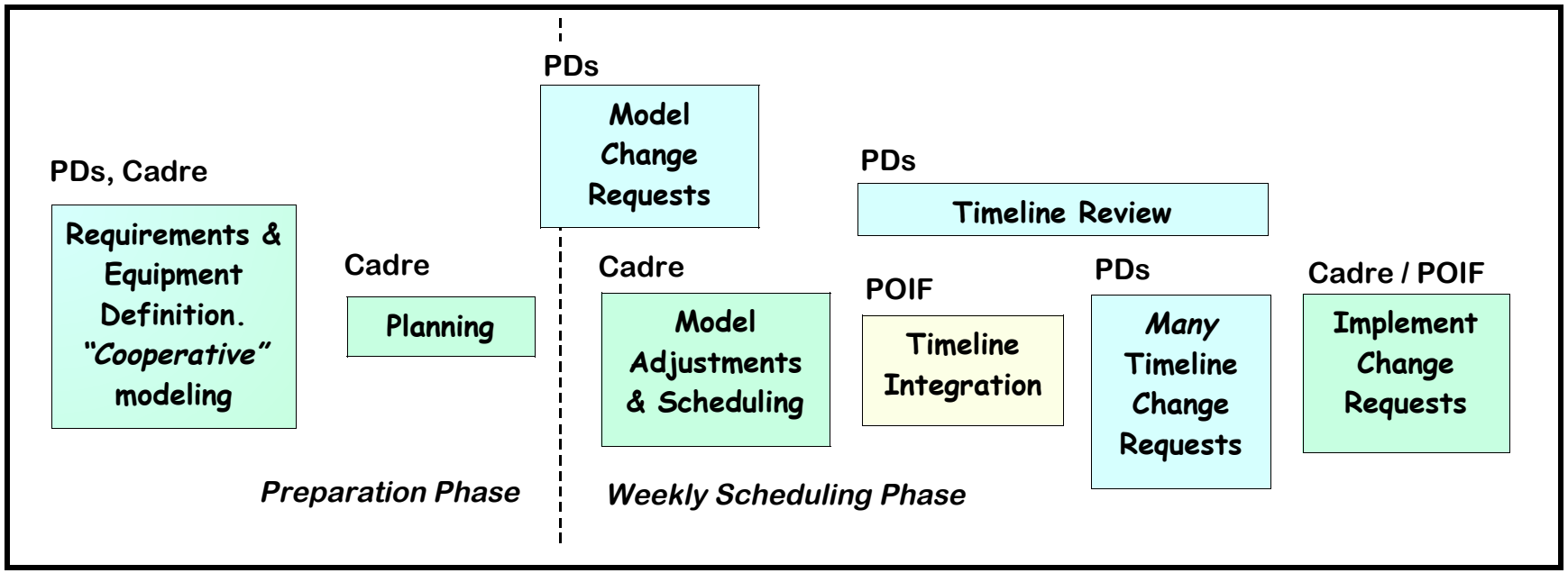
- **There are 3 phases to operations.**
  1. **Preparation for Scheduling**
  2. **Scheduling**
  3. **Execution**

- **Preparation occurs over a period of several weeks or months prior to the scheduling and execution. Scheduling and Execution occur on a rigid schedule as shown below.**



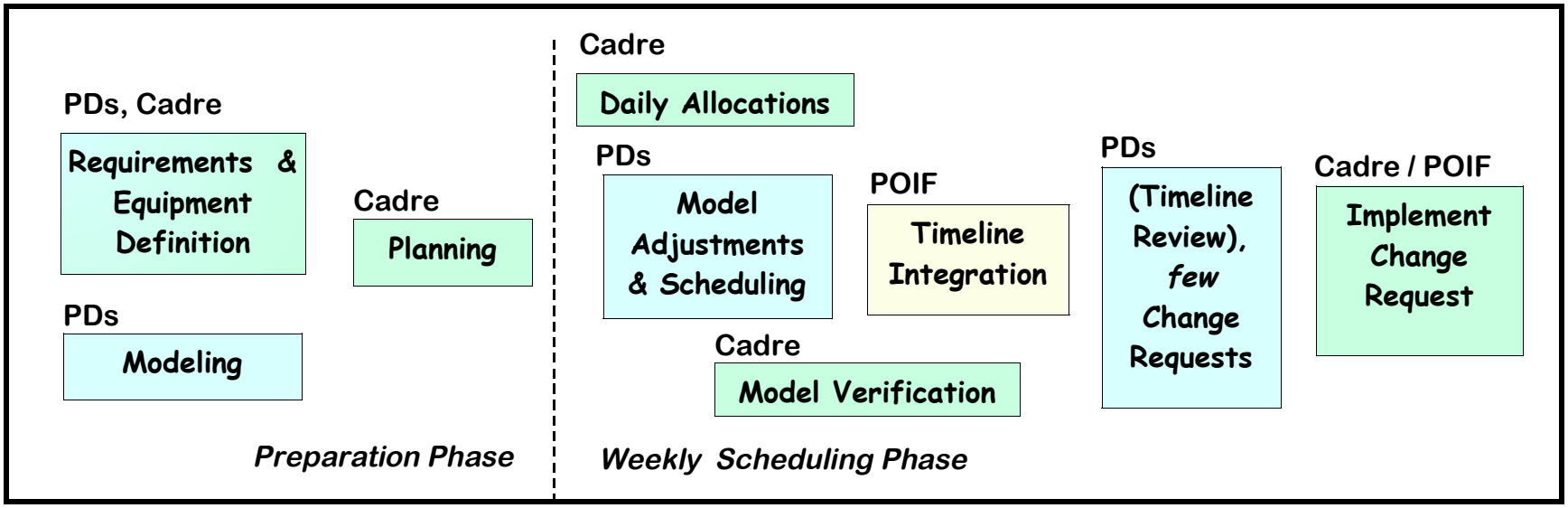
A ROSE-based Operations Concept (Current Concept for Comparison)

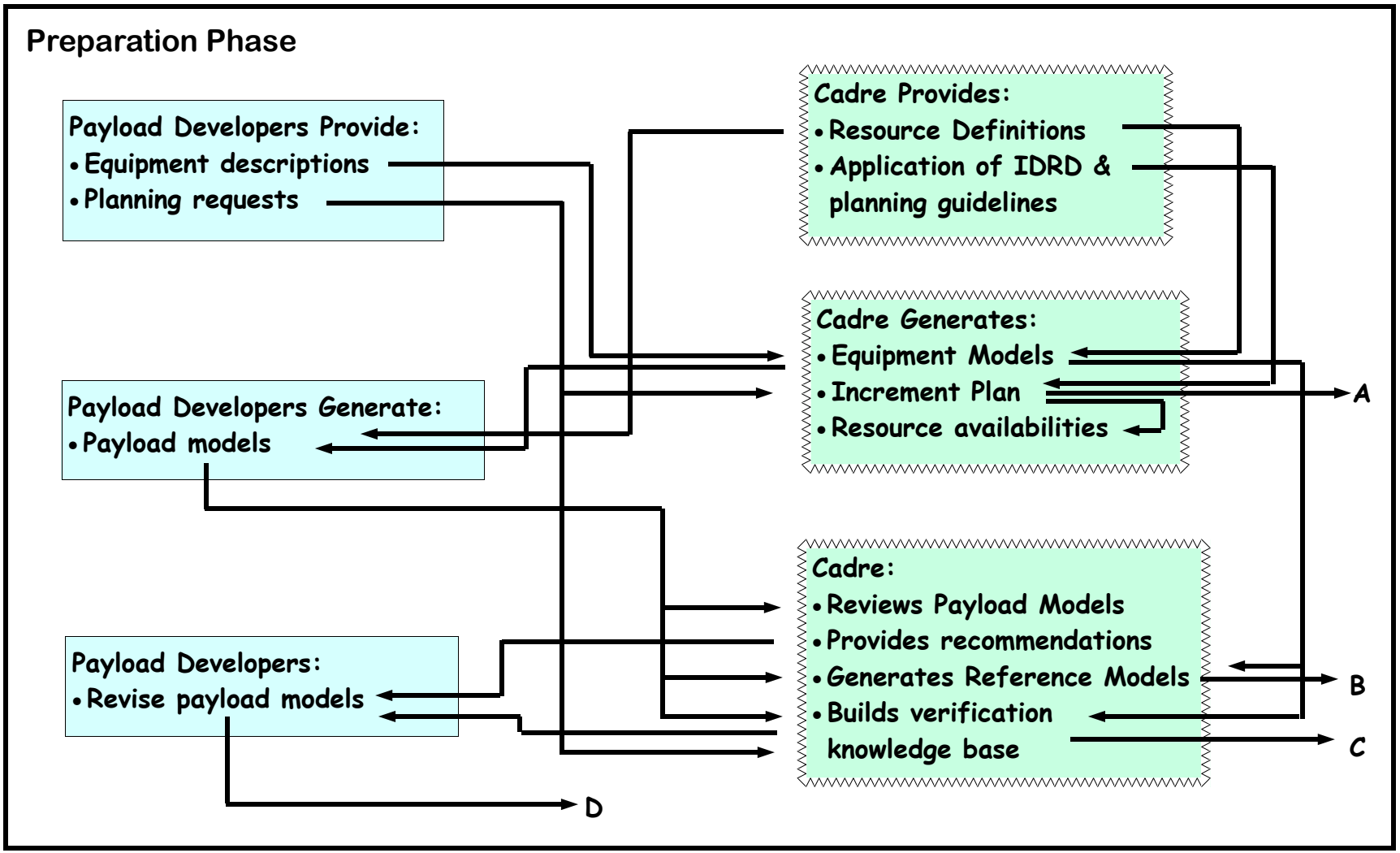
Modeling in the current concept is always a cooperative affair between the payload developer (PD) who knows his hardware and requirements and the scheduling cadre who knows how the scheduler works. There is no capability for the PD to acquire expertise in the scheduling engine because he has no direct feedback from the engine about the modeling process.

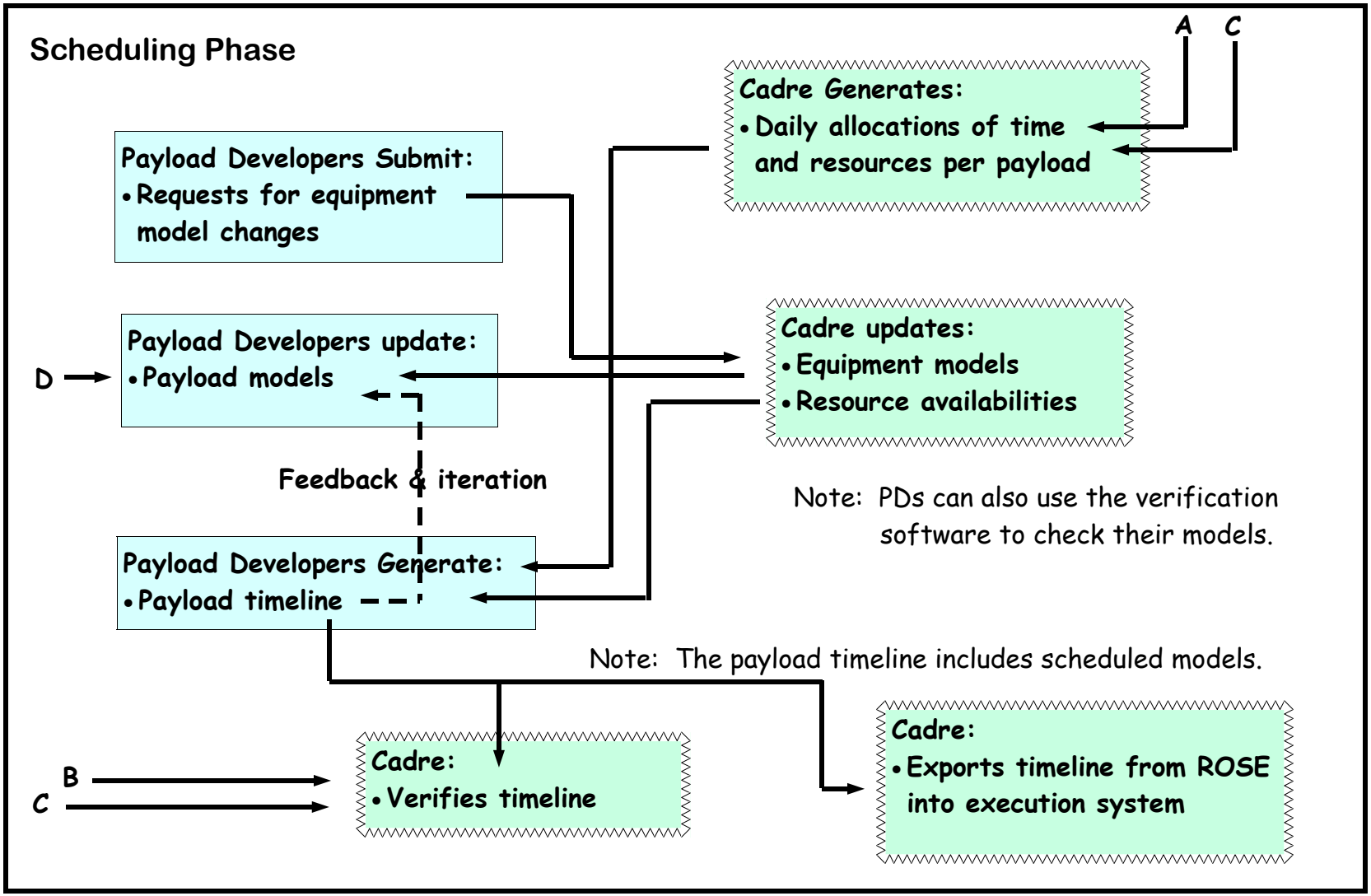


A ROSE-based Operations Concept (Proposed Concept)

Based on the increment plan produced during the preparation phase, and other information, the cadre generates daily allocations per payload for the week to be scheduled. The allocations are not profiles but are total usage limits of each resource during the planning week. Once the system is initialized with all the resource constraints, the *payload developers* use the ROSE system to produce a timeline. As always, producing a good timeline requires attempting to schedule a model, rejecting unacceptable results, tweaking the models, and trying again.




Preparation Phase
Payload Developers Provide:
• Equipment descriptions
• Planning requests
Payload Developers Generate:
• Payload models
Payload Developers:
• Revise payload models
Cadre Provides:
• Resource Definitions
• Application of IDRD & planning guidelines
Cadre Generates:
• Equipment Models
• Increment Plan
• Resource availabilities
Cadre:
• Reviews Payload Models
• Provides recommendations
• Generates Reference Models
• Builds verification knowledge base
A
B
C
D


Scheduling Phase
A
C
Cadre Generates:
• Daily allocations of time and resources per payload
Payload Developers Submit:
• Requests for equipment model changes
Payload Developers update:
• Payload models
D
Cadre updates:
• Equipment models
• Resource availabilities
Feedback & iteration
Note: PDs can also use the verification software to check their models.
Payload Developers Generate:
• Payload timeline
Note: The payload timeline includes scheduled models.
B
C
Cadre:
• Verifies timeline
Cadre:
• Exports timeline from ROSE into execution system

Timeline Validation

- **Timeline validation is done after the PDs have produced the timeline.**
- **Timeline validation only needs to validate the models.**
- ***Model validation is easy because all resources (except crew) are associated with equipment usage.***
- **Model validation is done by comparing the scheduled models to a reference set of models. (Reference models are built during the preparation stage.) It is only necessary to check:**
  - **Are all the equipment needed by this payload listed on the model?**
  - **Are the equipment modes correct?**
  - **Are the equipment modes in the correct sequence?**
  - **Are the durations within the allowed range?**

Timeline Validation = Model Validation. It is not necessary to validate the scheduling engine weekly.

Issues Rebutted
(full rebuttal in paper)

- **The Payload Developers Will "Fudge"**
- **The System Is Not Compatible With existing ISS support Software**
- **Payload Developers Don't Want To Participate**
- **"Daily" Allocations Are Too-Little / Too-Much**
- **Models Are Not Under Configuration Control**
- **The System Is Susceptible to Hacking**
- **First-Come, First-Serve Approach Is Unfair**
- **It's Too Hard For Payload Developers To Use**
- **There Is No Timeline Editor**

Benefits
(full discussion in paper)

- **Better Schedule**
- **Fewer Change Requests**
- **Less Cadre Time**
- **Less Cadre Burn-Out**
- **Available to the Crew**
- **Operates on Cheap Hardware**

Conclusion

The ROSE-based operations concept presented in this paper is an alternative concept that is operationally viable, is low cost, provides more autonomy to payload developers and is presented for future consideration in defining the long term operations concept for ISS payload operations.

- **The ROSE system is based on a proven modeling methodology that adequately and eloquently represents payload requirements.**
- **The architecture of the ROSE system has been developed and demonstrated. Research and development on the scheduling engine shows that it is feasible to develop a scheduling engine that matches the modeling methodology.**
- **The proposed process can easily be substituted for the current process, with reductions in required personnel.**

Most significantly, the ROSE-based concept allows the payload developer community to produce the timeline. They are the customers; they know what is required in the timeline better than anyone else; and the ROSE system will provide them with a convenient and robust tool to produce the timeline they want. At the same time, the ROSE-based concept provides adequate controls to ensure that the timeline meets programmatic constraints.